Vellore Institute of Technology

(Deemed to be University under section 3 of the UGC Act, 1956)

Airline Management System

- a J – component report

Name:	Regno:
Aileni Rohan Reddy	**19BCE2086**
Gajjala Niteesh Reddy	**19BCE0111**

A DBMS project on providing web application with proper user interface using database with proper frontend

Done under the

guidance of:

Faculty : Prof .Pradeep Kumar Roy sir

School Of Computer Science and Engineering

(SCOPE)

Vellore Institute of Technology

(Deemed to be University under section 3 of the UGC Act, 1956)

DECLARATION:

I hereby declare that the J Component report entitled “AIRLINES MANAGEMENT SYSTEM” submitted by me to Vellore Institute of Technology, Vellore in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology in Computer science and engineering is a record of Bonafide undertaken by me under the supervision of Prof. Pradeep Kumar Roy. I further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

ABSTRACT:

The web based “airline management system” project is an attempt to stimulate the basic concepts of airline reservation process. The system enables the customer to do the things such as search for airline flights for two travel cities on a specified date, choose a flight based on the details, reservation of flight and cancellation of reservation also provide user with options in prices from different travel agents. It also allows user to choose different agencies and provides user applications with all comforts required to them.

INTRODUCTION:

The purpose of the project is to develop an airline management system. This is a system that enable the customer to search for flights that are available between the two travel cities, namely the “Departure city” and “Arrival city” for a particular departure and arrival dates. The system displays all the flight details such as flight no, name, price and duration of journey, total no of seats, model etc.

After search, the system display list of available flights and allows customer to choose a particular flight. Then the system checks for the availability of seats on the flight. If the seats are available then the system allows the passenger to book a seat. Otherwise it asks the user to choose another flight.

To book a flight the system asks the customer to enter his details such as name, address, city, state, credit card number and contact number. Then it checks the validity of card through a secure portal and book the flight and update the airline database and user database. The system also allows the customer to cancel reservation, if any problem occurs.

It also gives an admin access in order to add/delete flights and manage user in database.

PROPOSED WORK:

We worked on three unique things in this project. We all know due to corona impact; every passenger is thinking of rides with safety measures so many flight agencies followed alternative seating. But if a family wants to go then this will cause some discomfort to them so in order to remove this the whole family is given an entire row so that we can increase economy which can be less dangerously.

Second thing as this is a service-based company, we totally relay on people satisfaction. We introduced automatic comment correction is automatic comment correction system, so that the employee who review the comments can read in an effective way and this may also reduce misunderstandings between employee and the customer.which means if any passenger who drops a comment with some mistake then this will correct those word automatically which helps to understand what the passenger wants to convey exactly.

The third thing is otp evaluation after booking a flight we may have to get an otp for making card transaction so that we will get an otp to the registered email and also confirmation mail and activation mail and forgot password otp is also sent.

Hardware/Software specification:

Hardware:
Any computer or mobile device to access.

Software:

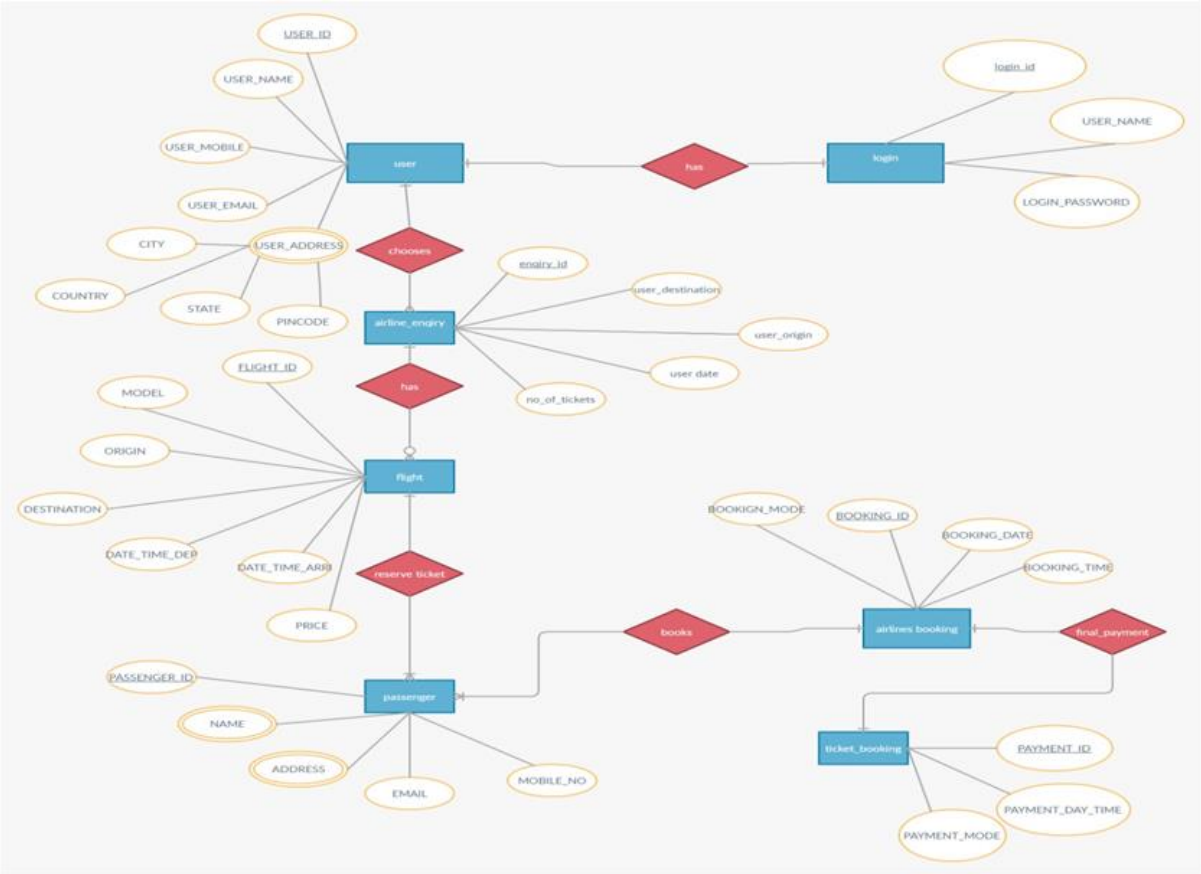
Backend: MySQL

Front end: HTML, Django Models, CSS

How we constructed a database and how we reduced redundancy?

Here is how our table structure is:

ER Diagram:



IMPLEMENTATION AND RESULTS:

```
| app_auto_correction
| app_bank
| app_city_code
| app_comment
| app_enquiry
| app_flight_info
| app_leg
| app_leg1
| app_login_log
| app_logout_log
| app_one_time_password
| app_passengers
| app_price
| app_shortcuts
| app_userinfo
| auth_group
| auth_group_permissions
| auth_permission
| auth_user
```

Table auth_group:

```
mysql> desc auth_group;
+-------+--------------+------+-----+---------+----------------+
| Field | Type         | Null | Key | Default | Extra          |
+-------+--------------+------+-----+---------+----------------+
| id    | int          | NO   | PRI | NULL    | auto_increment |
| name  | varchar(150) | NO   | UNI | NULL    |                |
+-------+--------------+------+-----+---------+----------------+
2 rows in set (0.01 sec)
```

This is for setting up a group

Table auth_group _permission:

```
mysql> desc auth_group_permissions;
+---------------+------+------+-----+---------+----------------+
| Field         | Type | Null | Key | Default | Extra          |
+---------------+------+------+-----+---------+----------------+
| id            | int  | NO   | PRI | NULL    | auto_increment |
| group_id      | int  | NO   | MUL | NULL    |                |
| permission_id | int  | NO   | MUL | NULL    |                |
+---------------+------+------+-----+---------+----------------+
3 rows in set (0.01 sec)
```

This table gives permissions for the bellowed groups for controlling wvwntd

Table auth_user:

```
mysql> desc auth_user;
+--------------+--------------+------+-----+---------+----------------+
| Field        | Type         | Null | Key | Default | Extra          |
+--------------+--------------+------+-----+---------+----------------+
| id           | int          | NO   | PRI | NULL    | auto_increment |
| password     | varchar(128) | NO   |     | NULL    |                |
| last_login   | datetime(6)  | YES  |     | NULL    |                |
| is_superuser | tinyint(1)   | NO   |     | NULL    |                |
| username     | varchar(150) | NO   | UNI | NULL    |                |
| first_name   | varchar(150) | NO   |     | NULL    |                |
| last_name    | varchar(150) | NO   |     | NULL    |                |
| email        | varchar(254) | NO   |     | NULL    |                |
| is_staff     | tinyint(1)   | NO   |     | NULL    |                |
| is_active    | tinyint(1)   | NO   |     | NULL    |                |
| date_joined  | datetime(6)  | NO   |     | NULL    |                |
+--------------+--------------+------+-----+---------+----------------+
11 rows in set (0.01 sec)
```

This table contains all the user data

Table app_userinfo:

```
mysql> desc app_userinfo;
+---------+------+------+-----+---------+-------+
| Field   | Type | Null | Key | Default | Extra |
+---------+------+------+-----+---------+-------+
| userid  | int  | NO   | PRI | NULL    |       |
| user_id | int  | NO   | UNI | NULL    |       |
+---------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

This table contain user_id for giving a specific if for user which is in one_to_one relationship with auth_user (as one user contain only one user_id)

Table app_bank:

```
mysql>  desc app_bank
    -> ;
+-----------------+--------------+------+-----+---------+-------+
| Field           | Type         | Null | Key | Default | Extra |
+-----------------+--------------+------+-----+---------+-------+
| account_id      | int          | NO   | PRI | NULL    |       |
| first_name      | varchar(100) | NO   |     | NULL    |       |
| last_name       | varchar(100) | NO   |     | NULL    |       |
| amount          | double       | NO   |     | NULL    |       |
| cvv             | int          | NO   |     | NULL    |       |
| expiration_date | varchar(5)   | NO   |     | NULL    |       |
| phone           | int          | NO   |     | NULL    |       |
| email           | varchar(254) | NO   |     | NULL    |       |
+-----------------+--------------+------+-----+---------+-------+
8 rows in set (0.14 sec)
```

This is a fake bank account table for reserving seats where while transfer an otp is sent to the app_bank email and transaction made to the flight company bank account

Table app_flight_info:

```
mysql> desc app_flight_info;
+-------------+------------+------+-----+---------+----------------+
| Field       | Type       | Null | Key | Default | Extra          |
+-------------+------------+------+-----+---------+----------------+
| id          | int        | NO   | PRI | NULL    | auto_increment |
| flight_id   | varchar(6) | NO   |     | NULL    |                |
| models_type | varchar(4) | NO   |     | NULL    |                |
| total_seats | int        | NO   |     | NULL    |                |
+-------------+------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

This is a flight body information but not flight information as one flight may contain may have more than route so this is uniqye for only flight but not for route

Table app_leg:

```
mysql> desc app_leg;
+---------------------------+--------------+------+-----+---------+-------+
| Field                     | Type         | Null | Key | Default | Extra |
+---------------------------+--------------+------+-----+---------+-------+
| leg_id                    | varchar(200) | NO   | PRI | NULL    |       |
| from_place                | varchar(150) | NO   |     | NULL    |       |
| to_place                  | varchar(150) | NO   |     | NULL    |       |
| duration                  | int          | NO   |     | NULL    |       |
| date_time_departure_stamp | datetime(6)  | NO   |     | NULL    |       |
| date_time_arrival_stamp   | datetime(6)  | NO   |     | NULL    |       |
| flight_id_id              | int          | NO   | MUL | NULL    |       |
| total_price               | double       | NO   |     | NULL    |       |
+---------------------------+--------------+------+-----+---------+-------+
8 rows in set (0.00 sec)
```

This table contains the leg_id which is a combination of datetimestamp of the the departure and flight id and

flight_id_id column is a has a foreign key for app_flight_info(above table)

Table app_leg1:

```
mysql> desc app_leg1
    -> ;
+-----------+--------------+------+-----+---------+----------------+
| Field     | Type         | Null | Key | Default | Extra          |
+-----------+--------------+------+-----+---------+----------------+
| id        | int          | NO   | PRI | NULL    | auto_increment |
| seats     | json         | NO   |     | NULL    |                |
| leg_id_id | varchar(200) | NO   | UNI | NULL    |                |
+-----------+--------------+------+-----+---------+----------------+
3 rows in set (0.01 sec)
```

This also contain the information about the seats available for the passenger of the particular route. Which is One_to_one for app_leg .

Table app_city_code:

```
mysql>  desc app_city_code;
+--------------+--------------+------+-----+---------+-------+
| Field        | Type         | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| IATA         | varchar(3)   | NO   | PRI | NULL    |       |
| city_name    | varchar(100) | NO   |     | NULL    |       |
| airport_name | varchar(150) | NO   |     | NULL    |       |
| country      | varchar(200) | NO   |     | NULL    |       |
+--------------+--------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

So this stores the city information like airport code city,country,which is also used in many tables as foreign key this table is made separately to reduce reductancy so it code of city change the we needed to change it for only that table only .

Table app_enquiry:

```
mysql> desc app_enquiry;
+-----------------+-------------+------+-----+---------+-------+
| Field           | Type        | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| enquiry_id      | varchar(6)  | NO   | PRI | NULL    |       |
| search_arri_city | varchar(50) | NO   |     | NULL    |       |
| search_depa_city | varchar(50) | NO   |     | NULL    |       |
| search_date_time | datetime(6) | NO   |     | NULL    |       |
| search_for_date | date        | NO   |     | NULL    |       |
| search_way_type | int         | NO   |     | NULL    |       |
| user_id         | int         | NO   | MUL | NULL    |       |
| no_of_pass      | int         | NO   |     | NULL    |       |
+-----------------+-------------+------+-----+---------+-------+
8 rows in set (0.01 sec)
```

This is used for history of the user as the user searches for the specific route then this store all the information like user,search date,arrival city,departure city.

Where user is the forignkey of app_userinfo (one user can make n enquiry) and an enquiry id is generated which is primary key for the table

Table app_login_log:

```
mysql> desc app_login_log;
+-----------------+-------------+------+-----+---------+----------------+
| Field           | Type        | Null | Key | Default | Extra          |
+-----------------+-------------+------+-----+---------+----------------+
| id              | int         | NO   | PRI | NULL    | auto_increment |
| login_id        | varchar(6)  | NO   |     | NULL    |                |
| login_date_time | datetime(6) | NO   |     | NULL    |                |
| user_id         | int         | NO   | MUL | NULL    |                |
+-----------------+-------------+------+-----+---------+----------------+
```

This is also same which is used to track user usage of website which store the details of the user login where user_id is the foreign key of userinfo .

Table app_logout_log:

```
mysql> desc app_logout_log;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| id          | int         | NO   | PRI | NULL    | auto_increment |
| logout_date | datetime(6) | NO   |     | NULL    |                |
| login_id_id | int         | NO   | MUL | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
3 rows in set (0.01 sec)
```

This is also same which is used to track user usage of website which store the details of the user logout where user_id is the foreign key of userinfo .

Table app_one_time_password:

```
mysql> desc app_one_time_password;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| otp         | int         | NO   |     | NULL    |       |
| start_time  | datetime(6) | NO   |     | NULL    |       |
| expiry_time | datetime(6) | NO   |     | NULL    |       |
| status      | varchar(1)  | NO   |     | NULL    |       |
| user_id_id  | int         | NO   | MUL | NULL    |       |
| otp_id      | varchar(4)  | NO   | PRI | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
6 rows in set (0.01 sec)
```

This is used when user request otp for changing password some time the user may press multiple time and we get multiple mails

So we used time stamp ordering protocol

Suppose if the presses the otp button for 4 time so we have 4 otp with status as T so for getting latest otp order otp by time so the last otp is the correct otp so make status of other 3 as F so when the customer enter the otp it gets authenticate

Table app_shortcut:

```
mysql> desc app_shortcuts;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| shortcut | varchar(100) | NO   | PRI | NULL    |       |
| abbri    | varchar(200) | NO   |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```

So while comment in moder age youngsters use shortcuts so while correcting it has two column 1)shortcut which contain shortcut word and the other abbri which means its full form

Table app_comment:

```
mysql> desc app_comment;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| comment_id   | varchar(5)  | NO   | PRI | NULL    |       |
| date_req     | date        | NO   |     | NULL    |       |
| org_comm     | longtext    | NO   |     | NULL    |       |
| exp1         | longtext    | YES  |     | NULL    |       |
| exp2         | longtext    | YES  |     | NULL    |       |
| exp3         | longtext    | YES  |     | NULL    |       |
| flight_id_id | int         | NO   | MUL | NULL    |       |
| user_id      | int         | NO   | MUL | NULL    |       |
| date_time    | datetime(6) | NO   |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
9 rows in set (0.01 sec)
```

So the website has a comment page which takes the comment from the user with all the information. And for colum for expected changes where user_id is foreign key of userinfo and flight_id_id is foreign key of flight_info.

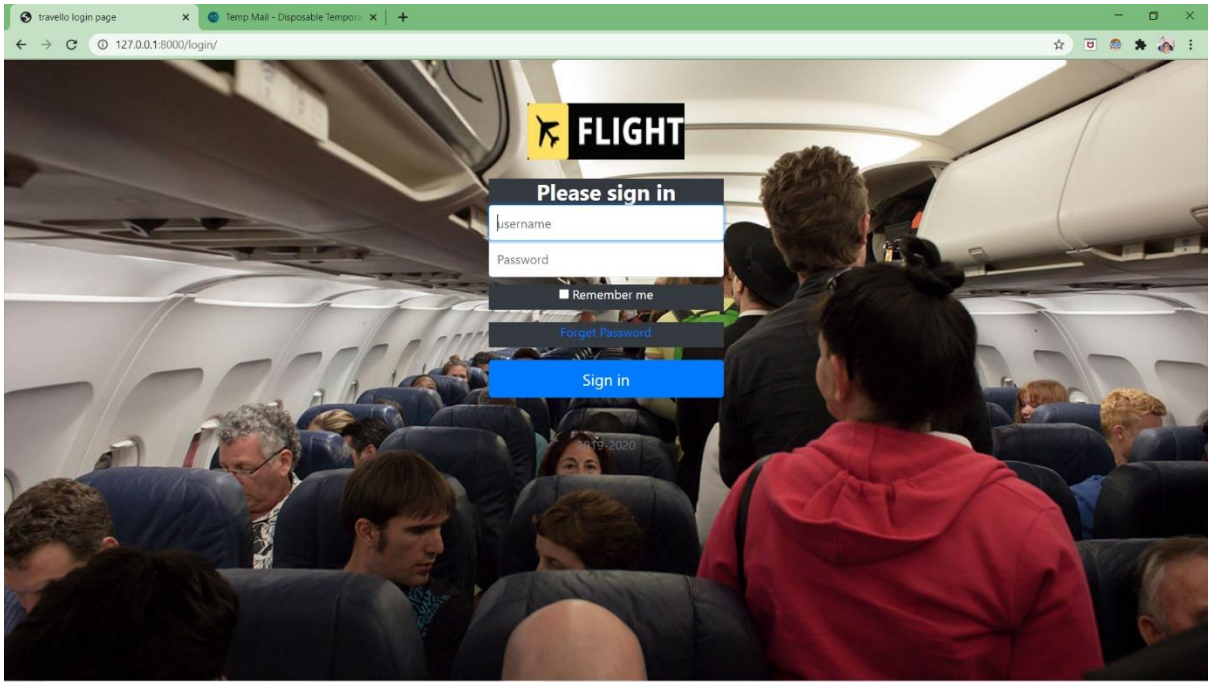
Table app_auto_correction:-

```
mysql> desc app_auto_correction;
+-----------+--------------+------+-----+---------+----------------+
| Field     | Type         | Null | Key | Default | Extra          |
+-----------+--------------+------+-----+---------+----------------+
| id        | int          | NO   | PRI | NULL    | auto_increment |
| title     | varchar(100) | NO   |     | NULL    |                |
| table1    | varchar(100) | YES  |     | NULL    |                |
| table2    | varchar(100) | YES  |     | NULL    |                |
| table3    | varchar(100) | YES  |     | NULL    |                |
| frequency | int          | NO   |     | NULL    |                |
+-----------+--------------+------+-----+---------+----------------+
6 rows in set (0.01 sec)
```

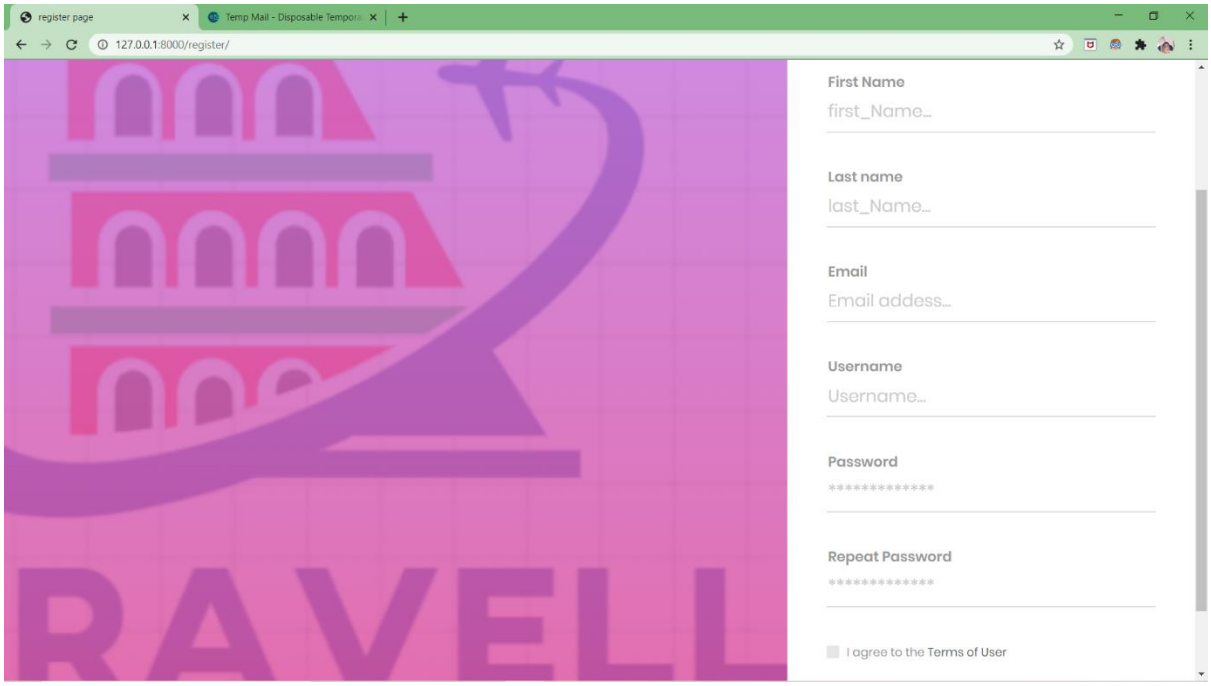
This is used for storing all the English words

Retrieving data from the database:

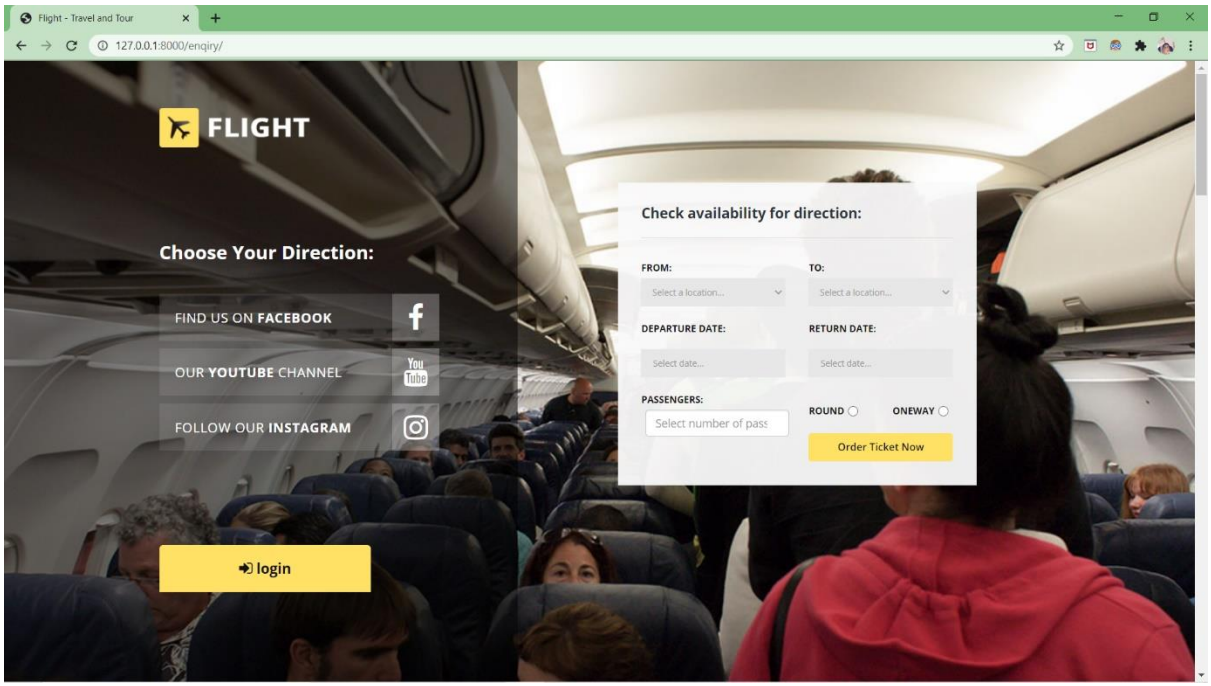
How an web page works:



This is the login page which uses the app_userinfo table to authticate the given user



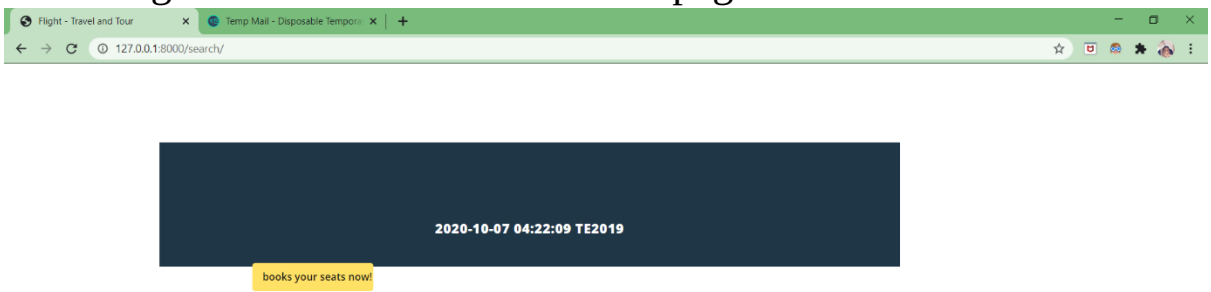
This is the user register page which write the information in the auth_user and app_userinfo tables



so when a person make some request in the above portal and then the request is been taken and been added to app_enquiry so that all our usage is tracked

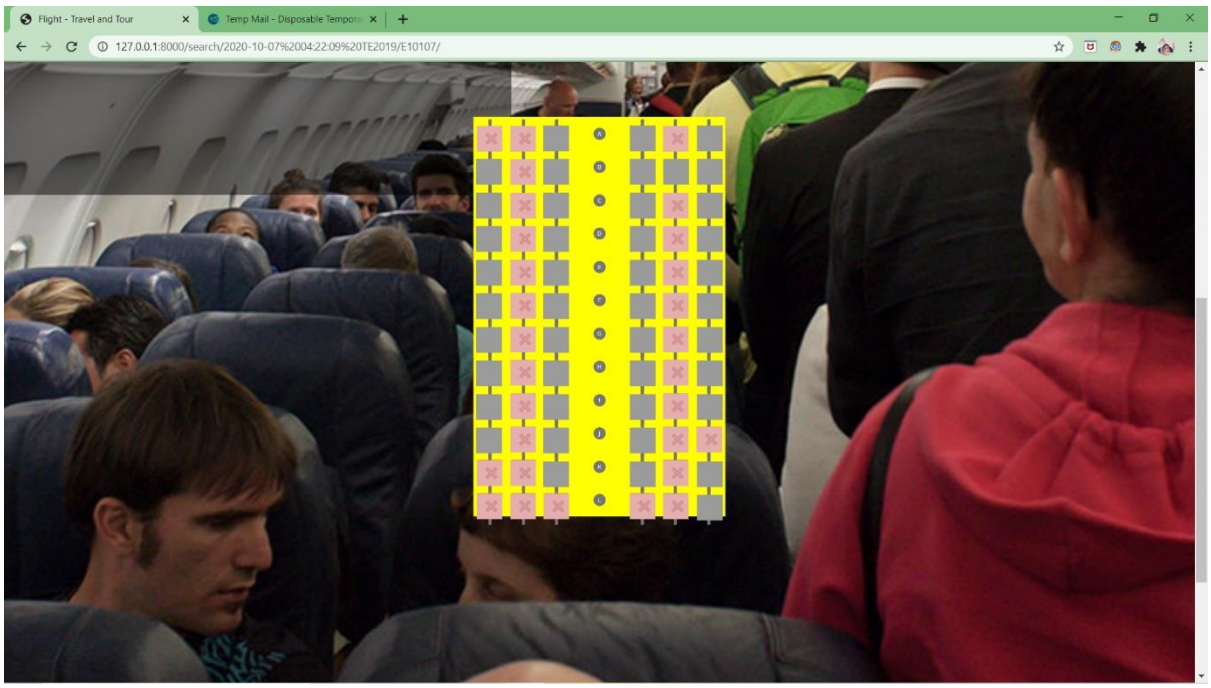
so when this happens it checks in app_leg for the flight for particular date and with arri_city and dep_city .

this image belows i s the result of that page:

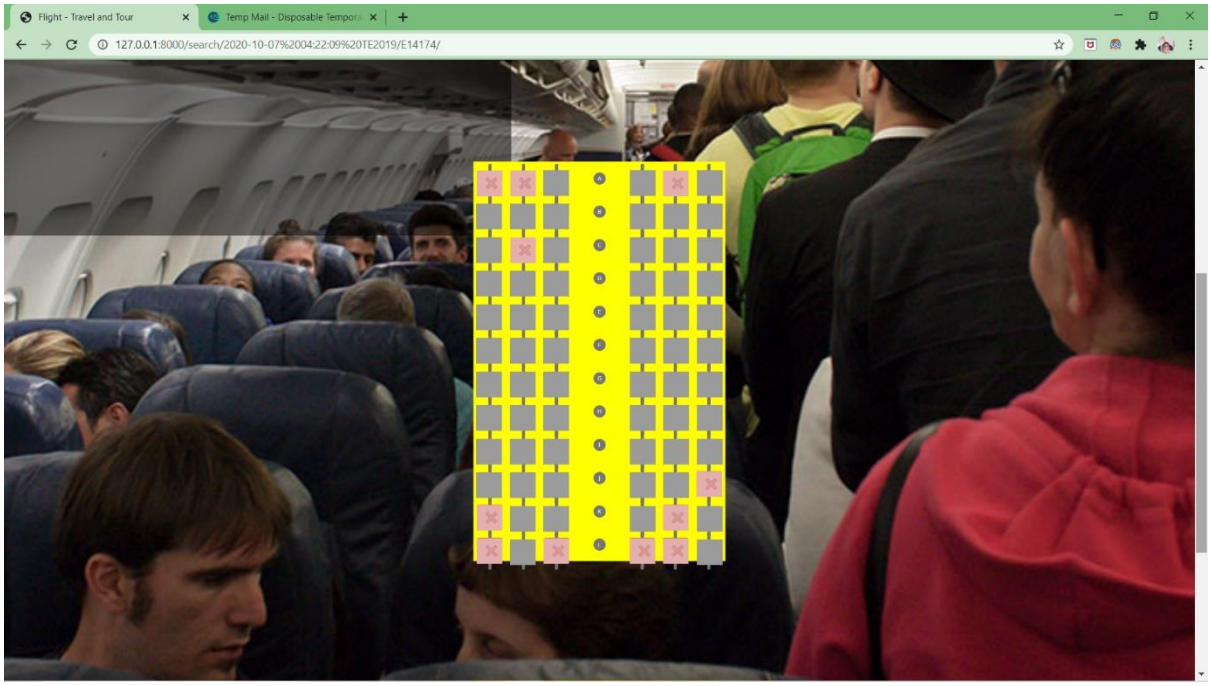


so, when we cliks book now there are two:

1)if enquiry no of passenger is 1 then show a different pattern.



2)if enquiry no is greater than 1 the we can remove the middle seats.

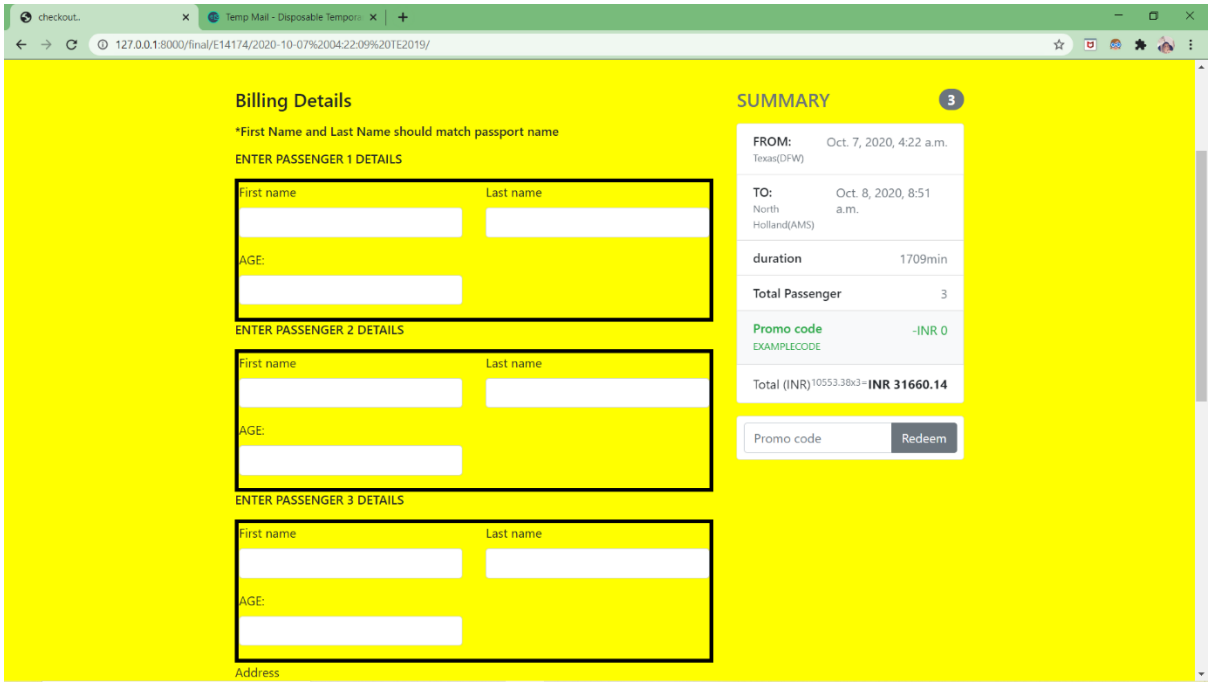


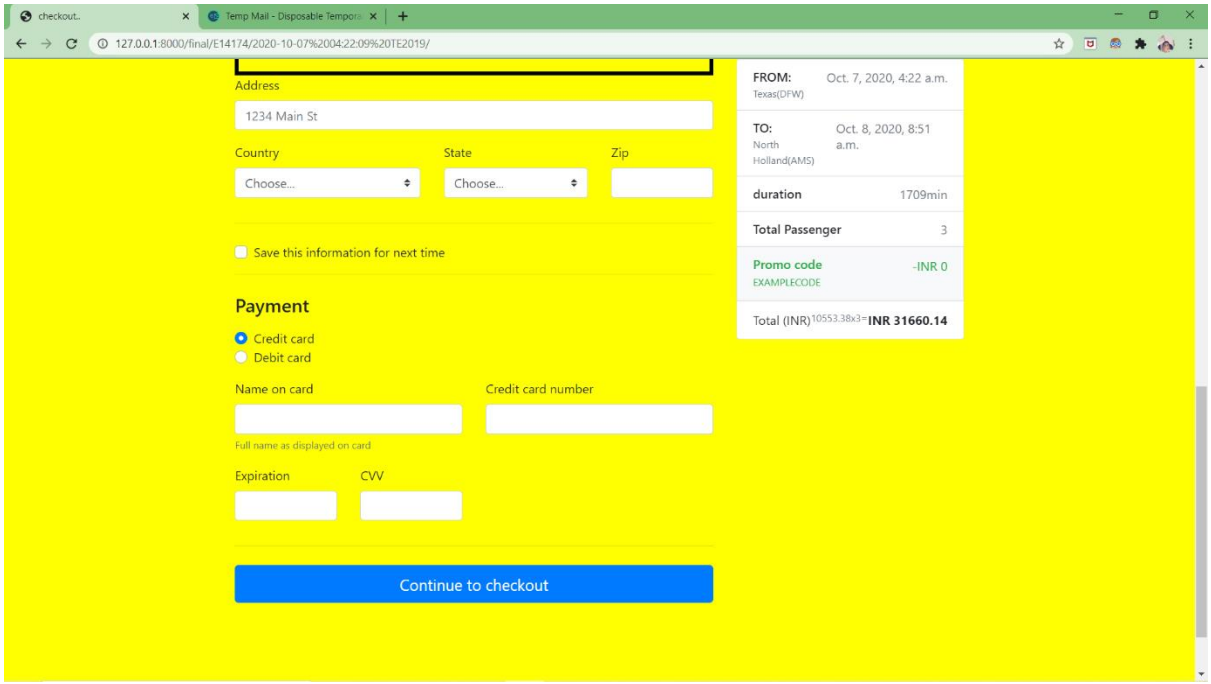
So how this page works

When we press for booking it check for app_enquiry for how many passengers we made a search for so pass is 1 then if render different html form but here connecting between the seats and the database is important so we store the seat booking information in app_leg1

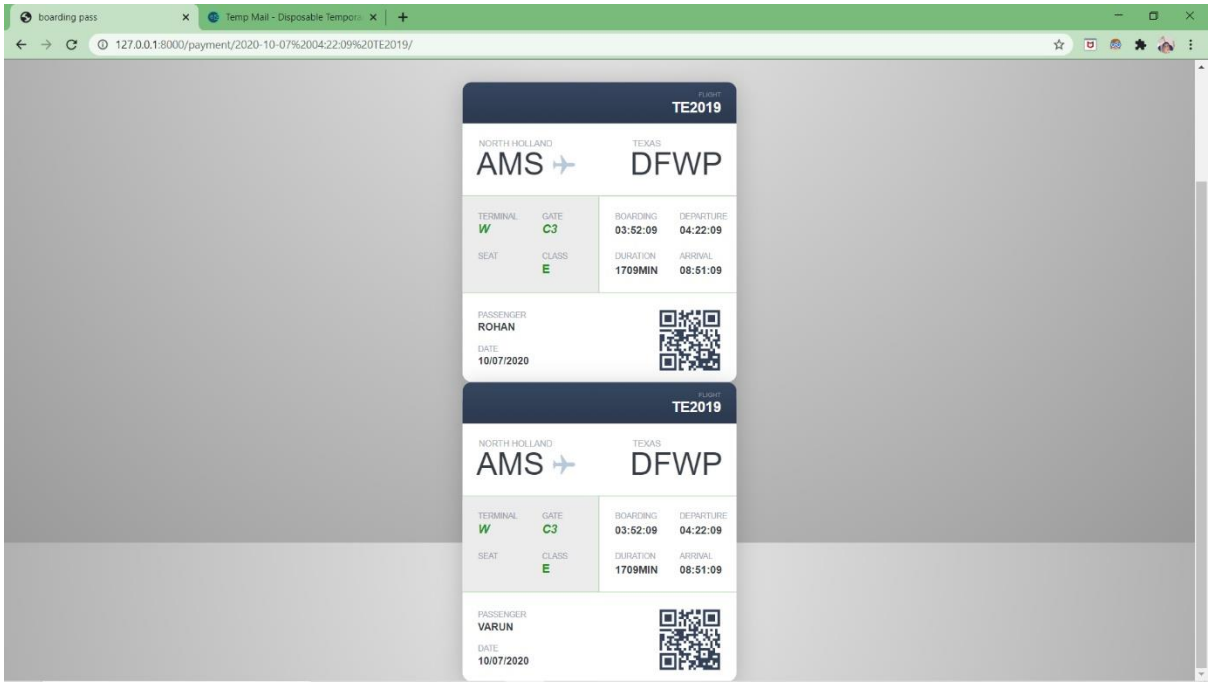
In a json object which is convenient for the browser to retrieve from the database so that we should be able to show booked seats as booked.

When we press confirm. Booking

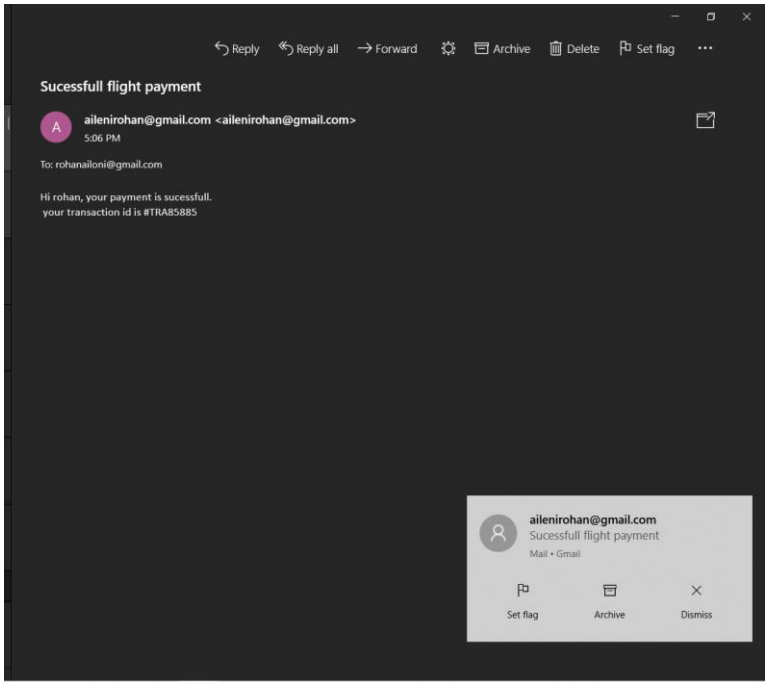




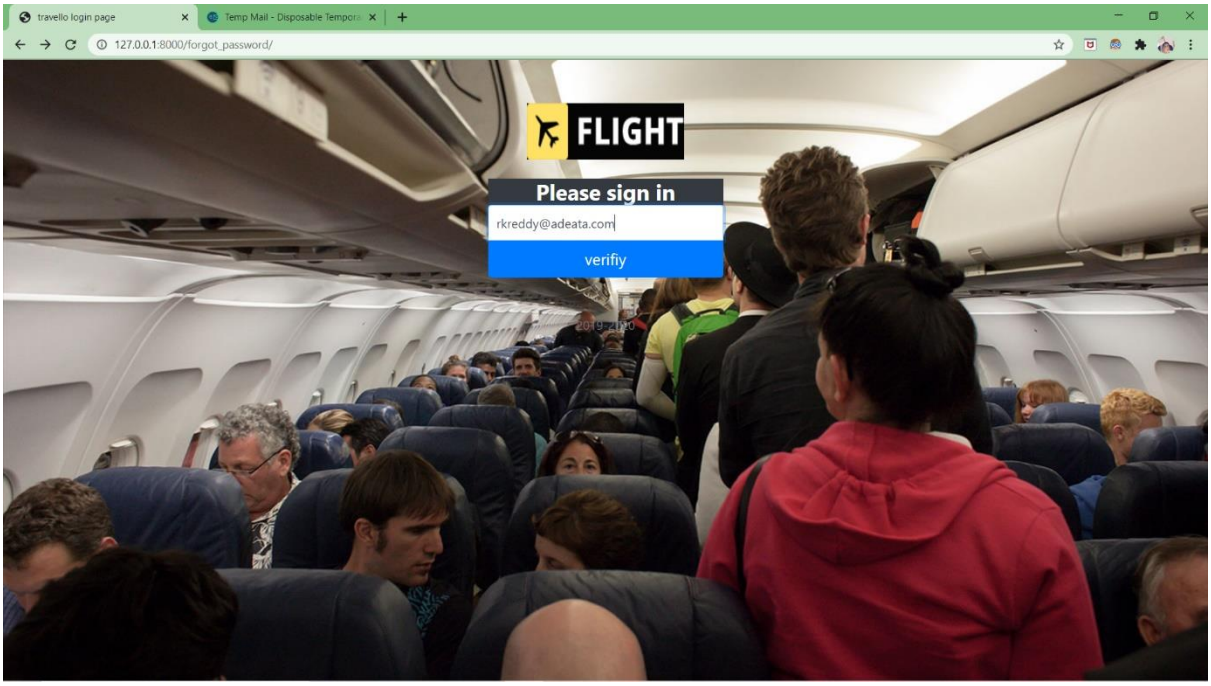
So we will get the form according to the number of passenger. And bill so in this form if we press card details it check the details in the card and send an otp to confirm that the bank transaction is made by original user. So this uses the app_Bank database to retive the card details and app_one_time_password to check the given otp is for right purpose. So if transaction is successful.



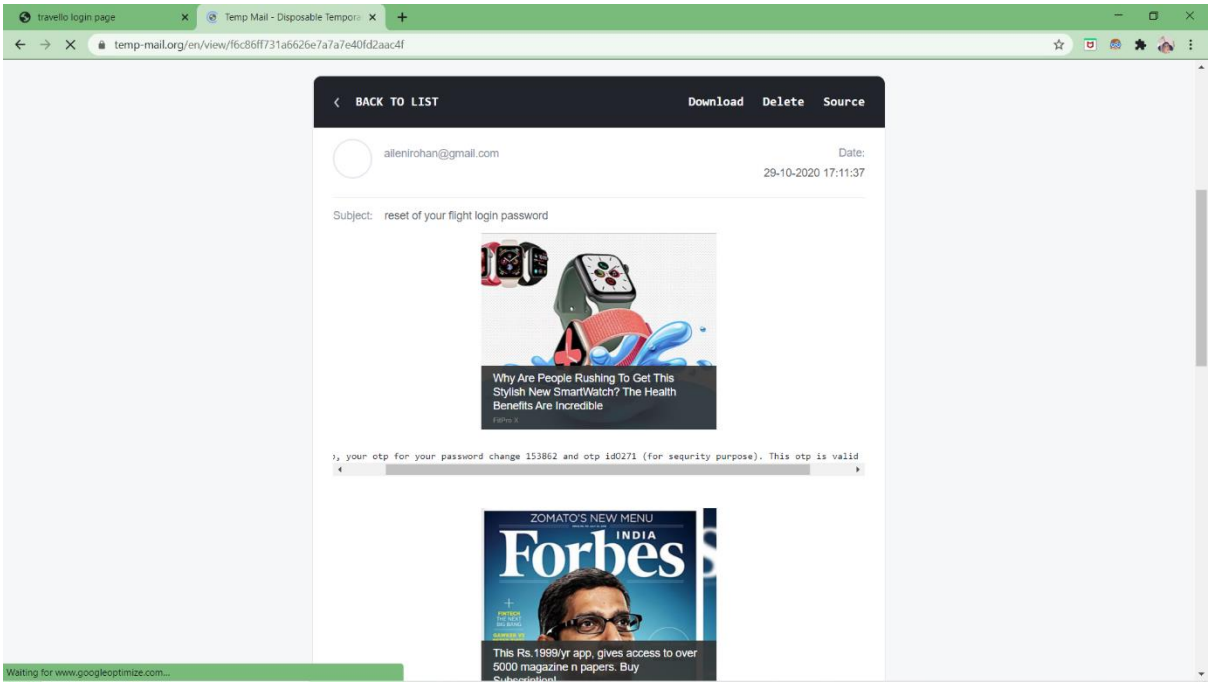
So you will get the boarding pass as well as you will get the mail about the transaction and its id for further checking boarding pass



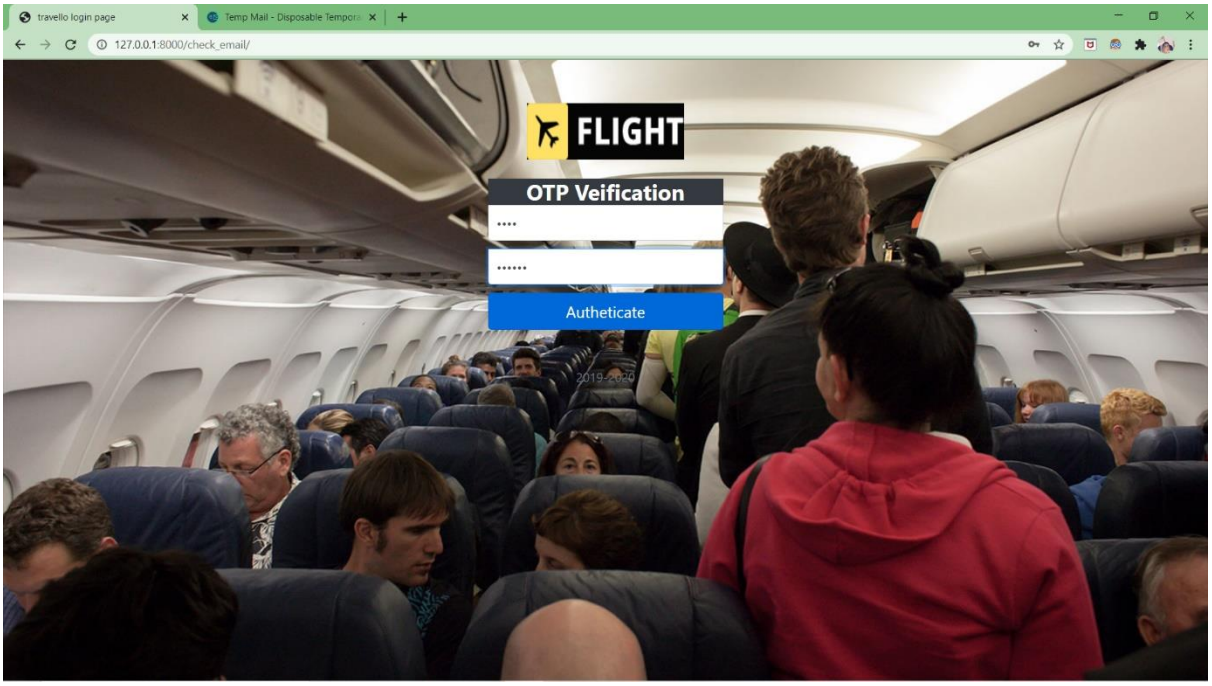
so if a user forgot his password then he has to press forgot password in the first



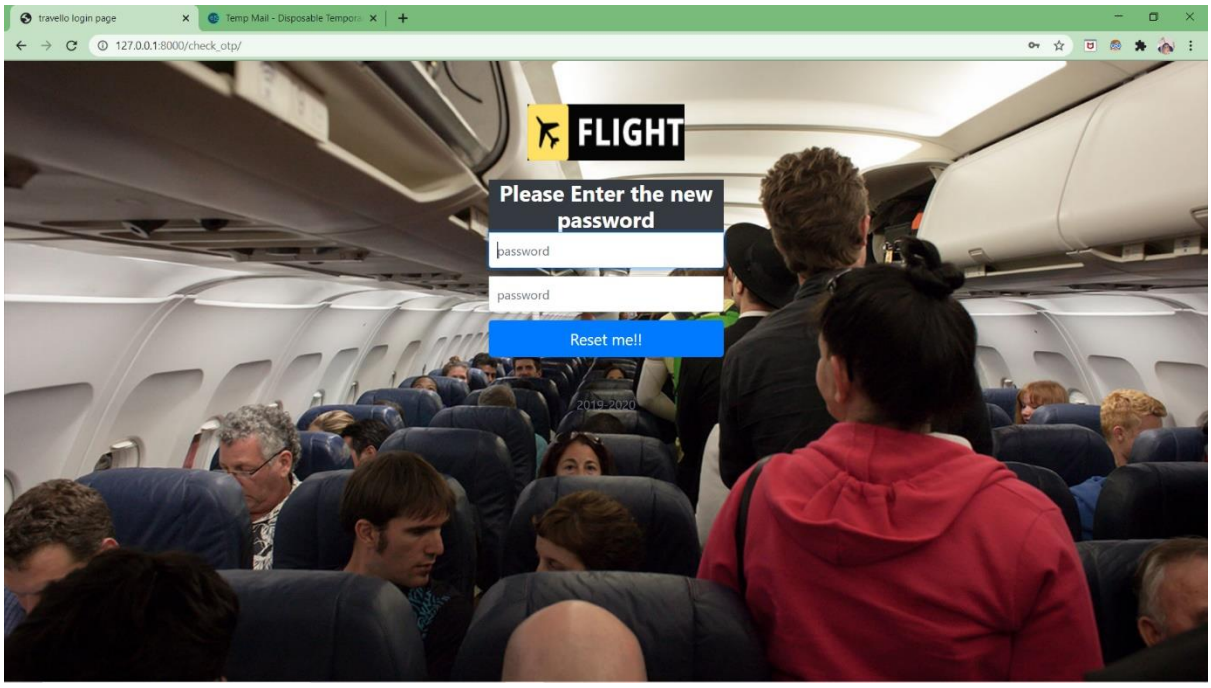
So you will get this page to enter the recovery email which checks the particular email in the auth_user table and sends an otp.



So you will get otp with id in the mail so enter in the below page.

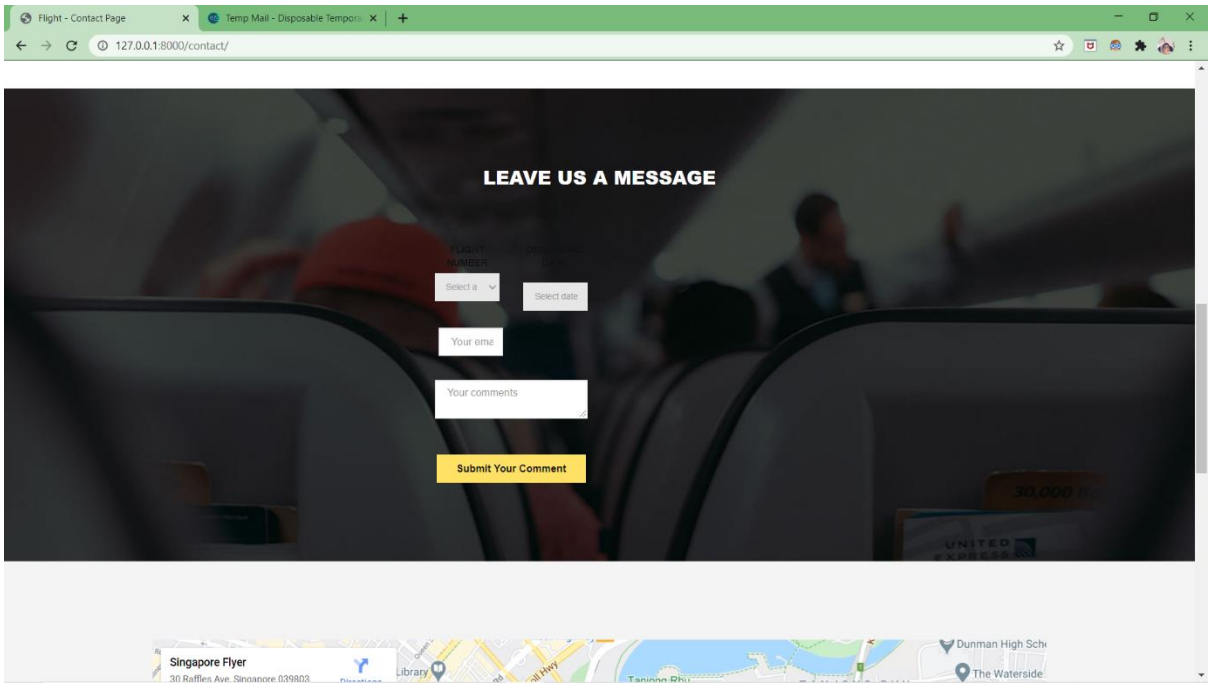


this takes otp and check in the database so that app_one_time_password_table so that it can authticate a person and take it to below form for changing password.

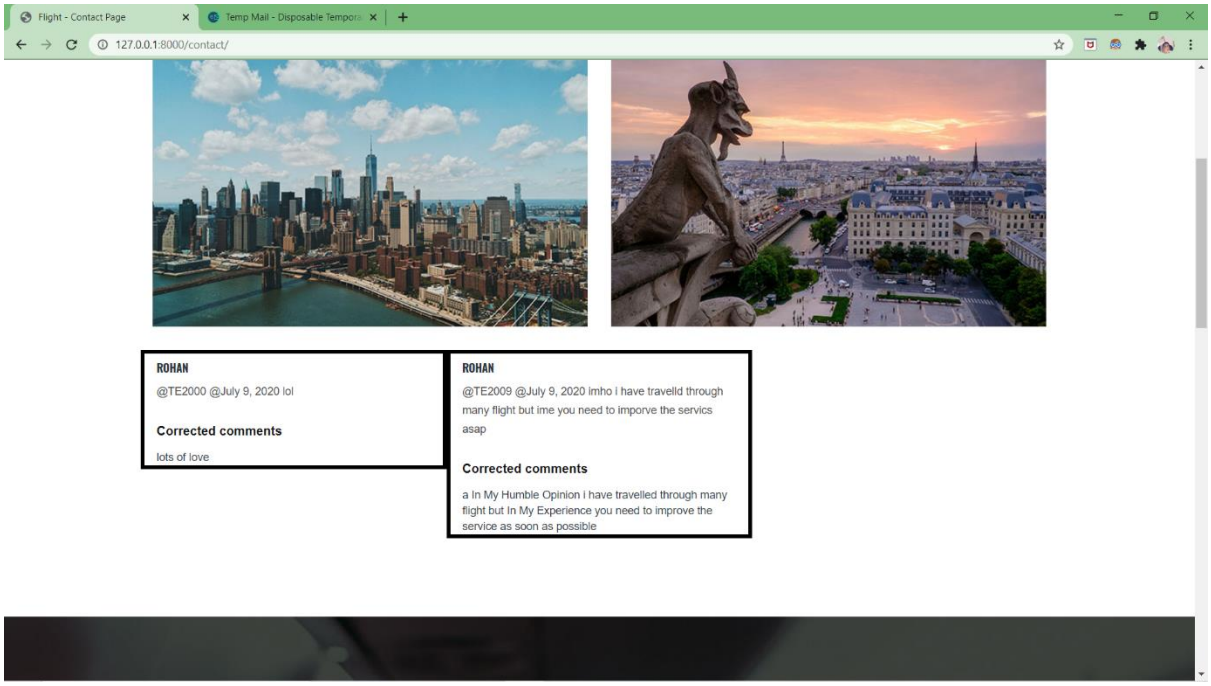


Where you can reset you password and it will redirect you to login page

So as this is service base company we surly rely on customer satisfaction so when the person so we have comment corrector for correcting the comment



That the place where we need to post the comment.



so how we used comment correction?

THE ALGORITHM USED:

```
Import re
From collections import counter
def words(text): return re.findall(r'\w+', text.lower())


WORDS = Counter(words(open('engimex.txt', "r", encoding='utf-8',
errors='ignore').read()))


def P(word, N=sum(WORDS.values())):
    return WORDS[word] / N


def correction(word):
    return max(candidates(word), key=P)


def candidates(word):
    return (known([word]) or known(edits1(word)) or
known(edits2(word)) or [word])


def known(words):
    return set(w for w in words if w in WORDS)
```

```
def edits1(word):
    letters = 'abcdefghijklmnopqrstuvwxyz'
    splits = [(word[:i], word[i:]) for i in range(len(word) + 1)]
    deletes = [L + R[1:] for L, R in splits if R]
    transposes = [L + R[1] + R[0] + R[2:] for L, R in splits if
len(R) > 1]
    replaces = [L + c + R[1:] for L, R in splits if R for c in
letters]
    inserts = [L + c + R for L, R in splits for c in letters]
    return set(deletes + transposes + replaces + inserts)


def edits2(word):
    return (e2 for e1 in edits1(word) for e2 in edits1(e1))
```

so what happens in this code so when a sentence is given it will be sliced into individual words and a file engimex.txt will be present which contain all the English words so the string sliced into may combinations using counter from collection module so when snd it looks up for all the words in the text file and if the sliced string has more resemblance then the missing letter is been added and

***BELOW code is used for implementation of above algorithm which is used in app/views.py in process(request) function**

```
comment1=request.POST.get('message')
comment1=comment1.split(" ")
print(comment1,1)
index_of_misspelled=[]
index_of_abbri=[]
for i in range(len(comment1)):
  if shortcuts.objects.filter(shortcut=comment1[i].lower().rstrip('\r\n')).exists():
    c=shortcuts.objects.get(shortcut=comment1[i].rstrip('\r\n'))
    index_of_abbri.append(i)
    comment1[i]=c.abbri

comment2=[]
for i in range(len(comment1)):
  if i not in index_of_abbri:
    comment2.append(comment1[i])
misspelled = spell.unknown(comment2)
misspelled1 = list(misspelled)

for i in range(len(misspelled1)):
```

```
if misspelled1[i] in comment1 :
      for j in range(len(comment1)):
         if misspelled1[i]==comment1[j]:
            index_of_misspelled.append(j)
            break


for i in range(len(misspelled1)):

   comment1[index_of_misspelled[i]]=spell.correction(misspelled1[i]).rstrip("\n ")

corrected_code=""
for i in comment1:
   corrected_code+=i+" "
print(corrected_code)
c=comment(comment_id=comment_id,user=u,flight_id=f1,date_req=date,org_comm=requ
est.POST.get('message'),exp1=corrected_code,date_time=datetime.datetime.now())
c.save()
return HttpResponseRedirect('/contact/')
```

CONCLUSION:

Simplicity is never simple. As we have seen in this project, the process of creating a user- friendly and straightforward platform that facilitates the administrator's job is one filled with complexity. From understanding user requirements to system design and finally system prototype and finalization, every step requires in-depth understanding and commitment towards achieving the objectives of the project.

Although the Airline management System is not fully integrated to the system and used on real time, the system prototype demonstrates easy navigation and data are stored in a systematic way. Overall, efficiency has improved and work processes simplified by using MySQL Databases. Although all the objectives have been met, the system still has room for improvement.